CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Mathematical Models, Heuristic and Exact Method for Order Picking in 3D-Narrow Aisles

Thomas Chabot
Leandro C. Coelho
Jacques Renaud
Jean-François Côté

June 2015

CIRRELT-2015-18

Document de travail également publié par la Faculté des sciences de l'administration de l'Université Laval, sous le numéro FSA-2015-005.

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

UNIVERSITÉ LAVAL · McGill · Concordia · ÉTS · UQÀM · HEC MONTRÉAL · POLYTECHNIQUE MONTRÉAL · Université de Montréal

Mathematical Models, Heuristic and Exact Method for Order Picking in 3D-Narrow Aisles

Thomas Chabot*, Leandro C. Coelho, Jacques Renaud, Jean-François Côté

Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Operations and Decision Systems, 2325 de la Terrasse, Université Laval, Québec, Canada G1V 0A6

Abstract. Order picking is one of the most challenging operations in distribution center management and one of the most important sources of costs. One way to reduce the lead time and associated costs is to minimize the total amount of work for collecting all orders. This paper is motivated by a collaboration with an industrial partner who delivers furniture and electronic equipments. We have modeled their 3D narrow aisle warehouse as a vehicle routing problem through a series of distance transformations between all pairs of locations. Security issues arising when working on narrow-aisles impose an extra layer of difficulty when determining the routes. Nevertheless, our approach yields an exact representation of all possible picking sequences. We have then solved large sets of instances reproducing realistic configurations using a combination of an heuristic and an exact algorithm, minimizing the total distance traveled for picking all items. Through extensive computational experiments, we identify for which aisle configurations our methods work best. We also compare our solutions with those obtained by the company order picking procedure, showing that significant improvements can be achieved by using our approach.

Keywords. Order-picking, narrow-aisle, warehousing, vehicle routing problem.

Acknowledgements. This research was partly supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grants RGPIN-2014-05764 and 01726-33. This support is gratefully acknowledged. We also thank the contact persons from our industrial partner.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

---

* Corresponding author: Thomas.Chabot@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2015

© Chabot, Coelho, Renaud, Côté and CIRRELT, 2015

1 Introduction

Distribution centers (DCs) play a central role in modern supply chains which are characterized by an increasing number of different products, or stock keeping units (SKUs), and by more frequent but smaller orders. Considering that order delivery within 24 hours is becoming a new industry standard, the performance of order picking operations in DCs are of critical importance. Because order picking is labor intensive for most warehouses, the design and control of warehouse order picking systems are highly strategic decisions [1, 13, 14, 16]. This paper is based on a collaboration with a Québec-based (eastern Canada) company which operates in the furniture and electronics industry. This industry is characterized by a very large number of SKUs since many products can be customized (type and color of woods and leathers, large variety of furniture fabrics, etc) and by large number of orders, each having few items. To efficiently deal with such an ordering pattern and product variety, the DC of our partner is organized in warehousing zones. Large electrical appliances are stacked directly on the floor; expensive electronic devices are stored in controlled access sections; finally, general furniture products, which represent about 70% of the total number of SKUs, are stored in narrow aisles with single-depth shelves on both sides, as depicted in Figure 1. Such mixed-width aisle configuration is useful to optimize both space utilization and order picking productivity [7].

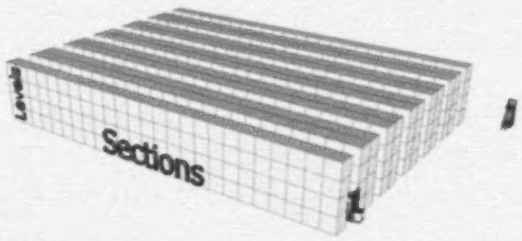

Figure 1: General overview of a distribution center with narrow aisles

Warehouse control presents numerous challenges, namely how to operate receiving, storing, order picking, and shipping [1, 3]. Storage and order picking are the two most

intensively studied operations. Storage addresses how the warehouse is divided and how space is allocated to products; order picking is related to how products are picked to fulfill customer orders. Some of the most important issues in order picking are batching procedures and routing methods [1, 5, 8, 9].

Standard large aisles warehouses have long been modeled and solved using routing methods. Goetschalckx and Ratliff [2] showed that the optimal aisle traversal can be modeled and solved as a shortest path problem. Single order picking in a general warehouse reduces to a standard traveling salesman problem (TSP). Ratliff and Rosenthal [11] showed that order picking in a rectangular warehouse is a special case of the TSP. Roodbergen and de Koster [12] developed an algorithm to find the shortest picking tour in a parallel aisle warehouse with a middle aisle. Despite these special cases, most realistic order picking problems are much more complex as they need to handle complicating constraints. Among these real-life constraints is the use of narrow aisles, which limits lift trucks circulation.

Narrow aisles are useful to maximize floor space utilization, and they facilitate the picking process as the picker has direct access to shelves on both sides of the aisle [4]. However, they require the use of special narrow aisle man-up turret lift truck, which must be driven according to some specific rules enforced by security reasons. In narrow aisles warehouses, traffic often becomes an important issue, but in our case traffic problems are eliminated by the DC practices according to which replenishments are performed during daytime while order picking and truck packing operations are performed at night. Also, each order picker is in charge of a specific number of aisles. These are some standard procedures used in this industry that make the problem at hand significantly different from other industries.

In this paper we describe the *Capacitated Narrow Aisle Order Picking Problem* (CNA-OPP) and show how to model it as a well-known vehicle routing problem (VRP) [17]. We derive a simple heuristic to reproduce the company decision-making procedure in order to accurately compute the cost of its current solutions. In addition, we develop an adaptive large neighborhood search (ALNS) heuristic as well as an exact branch-and-cut algorithm to solve realistic instances to optimality within very short computing times.

The remainder of the paper is organized as follows. In Section 2 we formally describe the CNA-OPP. Section 3 shows how to model this problem as a VRP taking into consideration the structure of the warehouse and security rules. Section 4 presents the algorithms we have developed to solve the problem. The results of extensive computational experiments of our algorithms and that of the company are detailed in Section 5, where we also provide a comparison of different layouts. Our conclusion are presented in Section 6.

2 Problem description

The warehouse can be formally described as being composed of narrow-aisles arranged with two single-depth racks having s sections long and t levels high. A set $\mathcal{P} = \{1, \ldots, m\}$ of m SKUs has to be picked up from a given aisle. The location of SKU i in the aisle can be defined as (x_i, y_i, z_i) where $x_i \in \{1, \ldots, s\}$, $y_i \in \{1, \ldots, t\}$, and $z_i \in \{1, 2\}$, where z_i represents the left or right rack of the aisle. We assume that x_1 is the first section of the aisle and x_s is the last one. The input/output (I/O) point of each aisle is modeled as being the position $(0, 0, 0)$. Each SKU i has a weight w_i in kilograms and a volume v_i in cubic feet. When performing a picking sequence, the maximum weight and volume that can be put on a pallet are W and V, respectively. All locations in the rack are identical, the horizontal distance between two consecutive sections is α_h, and the vertical distance between two consecutive levels is α_v. The parallel distance between left and right racks is equal to the aisle width and to that of the picking vehicle. Thus, the lateral distance is negligible and will be considered as zero. Because all m SKUs have to be picked up and given that the handling time is identical for all products, it can be ignored without loss of generality.

In order to ensure picker safety, two security rules must be respected. First, any horizontal lift truck movements, i.e., change of sections, must be performed at the ground level. The second constraint is that after having performed a pick, other truck movements should be in the backward direction to avoid products falling on the picker. Thus, if the pick of

product i is in section x_i, the next pick of product j must be in a section $x_j \leq x_i$. Note that it is allowed to pick at different levels and on both sides of the aisle because this does not imply a horizontal lift truck movement.

3 Distance matrix and mathematical formulation

We can define a distance matrix $\mathcal{D} = d_{ij}$, $i, j = 0, \ldots, m$, over the I/O point and all product locations. In order to respect safety constraints, $\mathcal{D}$ is defined as:

$$d_{ij} = \begin{cases} \alpha_h x_j + \alpha_v y_j & \text{if } x_i = 0 \quad (1a) \\ \alpha_v |y_i - y_j| & \text{if } x_i = x_j \quad (1b) \\ \alpha_h |x_i - x_j| + \alpha_v (y_i + y_j) & \text{if } x_i > x_j \quad (1c) \\ \infty & \text{otherwise.} \quad (1d) \end{cases}$$

Case (1a) computes the distances when starting from the I/O point to section x_j and level y_j. Case (1b) computes the distance between two locations in the same section as the sum of vertical distances, i.e., the difference between the levels. Note that this condition imposes a distance equal to zero from a location itself. Case (1c) is the more general case in which the picker changes sections, going forward from section x_i to x_j, with $x_i > x_j$. This distance is composed of two segments: the distance between the two sections ($\alpha_h |x_i - x_j|$) and the sum of the two vertical movements ($\alpha_v (y_i + y_j)$). Case (1d) prohibits backward movements from section x_i to x_j if $x_j > x_i$.

For example, consider a simple order with only two SKUs. The first pick i is at section $x_i = 5$, level $y_i = 4$ and on the left aisle, i.e., z_i=1, so at $(5, 4, 1)$, and the second pick j is at section $x_j = 2$, level $y_j = 3$ and right side, thus at $(2, 3, 2)$. Starting from the I/O point, the picker must first perform the pick at section 5 and level 4, go to the floor level, move back to section 2, and then go up to level 3. The first movement from the I/O point to product i incurs a distance of $(5\alpha_h + 4\alpha_v)$. Traveling from product i to product j

incurs a distance $(5-2)\alpha_h + (4+3)\alpha_v$. Returning from j to I/O point requires a distance of $(2\alpha_h + 3\alpha_v)$, for a total distance of $(10\alpha_h + 14\alpha_v)$. Note that if the picker decides to pickup the second item first, he will not be able to move forward to pickup the first one in the same trip. In this case, two trips from I/O point are required with a total distance of $(14\alpha_h + 14\alpha_v)$.

3.1 Mathematical model

The CNA–OPP consists of determining the order picker routes, starting and ending at the I/O point, such that all products are picked and the traveled distance is minimized while respecting capacities and safety constraints. It is modeled over a graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ where $\mathcal{V} = \{0, 1, \ldots, m\}$ is the set of vertices in the aisle with a picking demand, composed of the m location nodes $\mathcal{V}' = \{1, 2, \ldots, m\}$ assigned to specific requests with weight w_j and a volume v_j. We define the arc set $\mathcal{A} = \{(i,j) : i, j \in \mathcal{V}, i \neq j, i = 0, \ldots, m, j = 1, \ldots m\}$ and distance d_{ij} associated with arc (i,j).

In order to define a transportation plan that respects the practical constraints of this problem and minimizes the total work of the picker, we define the following decision variables. Let x_{ij} be a binary variable equal to 1 if arc (i,j) is used, and k be an integer variable indicating the number of routes (picking tours) needed. The CNA-OPP can then be modelled as follows:

$$\text{minimize} \sum_{(i,j)\in\mathcal{A}} d_{ij}x_{ij} \tag{2}$$

subject to

$$\sum_{j\in\mathcal{V}} x_{0j} = k \tag{3}$$

$$\sum_{i\in\mathcal{V}} x_{i0} = k \tag{4}$$

$$\sum_{j \in \mathcal{V} \setminus \{i\}} x_{ij} = 1 \qquad \forall i \in \mathcal{V} \tag{5}$$

$$\sum_{i \in \mathcal{V} \setminus \{j\}} x_{ji} = 1 \qquad \forall i \in \mathcal{V} \tag{6}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - r(s) \qquad S \subseteq \mathcal{V}, |S| \geq 2 \tag{7}$$

$$k \geq 0 \text{ and integer} \tag{8}$$

$$x_{ij} \in \{0, 1\}. \tag{9}$$

The objective function (2) minimizes the total distance traveled. Constraints (3) and (4) ensure that the same number of routes will leave and return to the I/O point. Constraints (5) and (6) are degree constraints and ensure that all products will be picked up exactly once, while (7) simultaneously forbids subtours and ensures that the volume capacity is respected. Here, $r(s) = max\left\{\left\lceil \frac{\sum_{j \in S} v_j}{V} \right\rceil, \left\lceil \frac{\sum_{j \in S} w_j}{W} \right\rceil\right\}$. Movement restrictions do not need to be imposed through constraints as they will be avoided due to the values from the distance matrix.

3.2 Arc reductions

We now present three procedures to remove several arcs from the graph and significantly decrease its size.

The first procedure is due to the restriction of the picker to move from lower sections to higher sections. For security reasons, all arcs from a lower section to a higher section can be removed as these paths are not feasible.

The second procedure considers products located in the same section and at the same level, but at different sides. Since the picker can pick the items in any order, these create symmetric solutions. We can thus remove all arcs (i, j) with $i < j$.

The last procedure considers products from the same section but at different levels. Recall that the picker always starts and ends a section at ground level. This structure enables

the removal of arcs between products in the same section. Let S be a set of products located on different levels of a same section. Among all paths visiting all the products of S only two are of minimal distance. The first path collects products from the lowest level to the highest level. The second one does the opposite, collecting products from the highest level to the lowest one. For example, if three products are located on levels 3, 7 and 11 then paths (3, 7, 11) and (11, 7, 3) are clearly the least costly. Other paths such as (7, 3, 11) will require additional vertical movements. To remove all more expensive paths, and at the same time subtours within the same section, all arcs from a product located in a lower level to a higher level in the same section can be removed.

To summarize our rules, arc (i, j) with $i \in V \setminus \{0\}$ is removed if any of the following applies:

1. $x_i < x_j$;

2. $x_i = x_j$ and $y_i < y_j$;

3. $x_i = x_j$ and $y_i = y_j$ and $i < j$.

Proposition 1. *The use of all these procedures eliminate all cycles in the graph and thus prevent having solutions with subtours.*

Proof By contradiction, suppose all procedures were applied and at least one cycle is present. It means there exists a path from a product i located in a lower section, or at lower level, or having a lower index than another product j. However, because all procedures were applied, no arcs exist from i to k where k is in a farter section, or at a higher level or having a higher index. Consequently, the graph does not contain any cycles. □

4 Solution methods

Since the CNA-OPP can be modeled as a CVRP over an asymmetric graph, we have implemented three different algorithms for its resolution. In Section 4.1 we describe the order picking method used by our industrial partner. Section 4.2 presents an adaptive large neighborhood search metaheuristic [15], and in Section 4.3 we present a branch-and-cut algorithm.

4.1 Actual picking method used by the company

In order to compare our results with those obtained by our industrial partner, we have implemented an heuristic that reproduces their current picking method. For each picking route, they start with the product in the farthest section at the highest level. Then, they add the next feasible product from the next highest level in the same section. If no more products can be feasibly picked in the same section they go back to the next farthest section where a product can be feasibly picked. In this way they always choose the product in the farthest section at the highest level, which respects to security movement constraints. The pseudocode for this set of rules is presented in Algorithm 1.

First, we use a list L of products which is ordered by sections, and for the same section it is sorted by levels. For a same level, ties between left and right aisle is broken arbitrarily (line 1). Then, create a new route R, and start by picking the farthest product $L[1]$, i.e., the one at the farthest section at the higher level (lines 3 and 4). The route is completed by picking the next closest item or until the capacity constraints are reached, always respecting the safety movements restrictions (lines 6 to 10). When an item is added to a route, it is removed from the list L and all following items in L are moved up once. When the route is filled, another route is created and initialized again with the farthest item from the remaining items from L. This approach is optimal if the total volume and weight of all the products to be picked up is less or equal to the capacity V and W of the picking vehicle.

Algorithm 1 Picking method used by the company

1: Sort the set of requests L such that the following two conditions hold for all $i, j \in L$
$x_{L[i]} \geq x_{L[j]}$ and if $x_{L[i]} = x_{L[j]}$ then $y_{L[i]} \geq y_{L[j]}$
2: **while** L is not empty **do**
3: Create a new picking route R with a total volume $D = 0$ and total weight $P = 0$
4: Initialize route with $L[1]$
5: $D = D + v_{L[1]}$ and $P = P + w_{L[1]}$ and $L = L \setminus L[1]$
6: **for** request i in the sorted L **do**
7: **if** $D + v_{L[i]} < V$ and $P + w_{L[i]} < W$ **then**
8: Add request $L[i]$ to the route R
9: $D = D + v_{L[i]}$ and $P = P + w_{L[1]}$ and $L = L \setminus L[i]$
10: **end if**
11: **end for**
12: **end while**

4.2 Adaptive large neighborhood search

We present an implementation of an ALNS heuristic for our problem, widely based on Ropke and Pisinger [15]. The ALNS is composed of a set of simple destruction and reconstruction heuristics in order to find better solutions at each iteration. An initial solution can be considered to speed up the search and the convergence of the algorithm. We have implemented a fast sequential insertion heuristic, which performs a greedy search for the best insertion for one product at a time.

The ALNS selects one of many destroy and repair operators at each iteration. We have implemented three destroy and two repair operators. Our destroy operators include the Shaw removal [18], the worst removal, and a random removal. Our repair operators include a greedy parallel insertion and a k-regret heuristic [10] . Each operator is selected with a probability that depends on its past performance and a simulated annealing acceptance criterion is used.

The ALNS is ran for 50 000 iterations of destroy-repair operators. After every 100 iterations, the weight of each operator is updated according to its past performance. Initially, all the operators have the same weight. A sketch of our ALNS is provided in Algorithm 2 and for further details see Ropke and Pisinger [15].

Algorithm 2 Adaptive large neighborhood search

1: Create an initial solution s using sequential insertion operators : $s' = s$
2: D : set of removal operators, I : set of repair operators
3: Initiate probability ρ^d for each destroy operator $d \in D$ and probability ρ^i for each repair operator $i \in I$
4: **while** stop criterion is not met **do**
5: Select remove ($d \in D$) and insert ($i \in I$) operators using ρ^d and ρ^i
6: Apply operators and create s^t
7: **if** s^t, is accepted by the simulated annealing criterion **then**
8: $s = s^t$
9: **end if**
10: **if** $cost(s) < cost(s')$ **then**
11: $s' = s$
12: **end if**
13: Update ρ^d and ρ^i
14: **end while**
15: return s'

4.3 Branch-and-cut algorithm

We have implemented a classic branch-and-cut algorithm in which valid linear inequalities are used as cutting planes to strengthen a linear programming relaxation at each node of a branch-and-bound tree. Constraints (7) are initialy relaxed and only added to the branch-and-bound tree if they are found to be violated.

The whole model can be solved by feeding it directly into an integer linear programming solver to be solved by branch-and-bound if the number of rounded capacity inequalities (7) is not excessive. However, for instances of realistic size, the number of rounded capacity constraints (7) is too large to allow full enumeration and these must be dynamically generated throughout the search process. The exact algorithm we present is a branch-and-cut scheme in which the rounded capacity inequalities constraints are generated and added into the program whenever they are found to be violated. It works as follows. At a generic node of the search tree, a linear program containing the model with a subset of the subtour elimination constraints and relaxed integrality constraints solved, a search for violated inequalities is performed, and some of these are added to the current program which is then reoptimized. This process is reiterated until a feasible or dominated solution is reached, or until there are no more cuts to be added. At this point, branching on a fractional variable occurs. We provide a sketch of the branch-and-cut scheme in Algorithm 3.

5 Computational results

The algorithms described in Section 4 were implemented in C++. The branch-and-cut uses the CVRPSEP library [6] and IBM CPLEX Concert Technology 12.5 as the branch-and-bound solver. All computations were executed on machines equipped with two Intel Westmere EP X5650 six-core processors running at 2.667 GHz, and with up to 48 GB of RAM installed per node running the Scientific Linux 6.3. We have first solved each instance using the ALNS algorithm. This solution was then used to start the branch-and-cut, to which a time limit of 7200 seconds was imposed.

In order to evaluate the performance of our algorithms and that of the company, we have generated instances inspired from real warehouse configurations. First we have generated five types of aisle configurations as presented in Table 1. The second and the third columns show respectively the number of sections and levels present in an aisle. We also show the

Algorithm 3 Branch-and-cut algorithm

Subproblem solution: Solve the LP relaxation of the current node
Termination check
if there are no more nodes to evaluate **then**
    Stop
else
    Select one node from the branch-and-cut tree
end if
while solution of the current LP relaxation contains subtours **do**
    Identify connected components with CVRPSEP [6]
    Add violated subtour elimination constraints
    Subproblem solution. Solve the LP relaxation of the current node
end while
if the solution of the current LP relaxation is integer **then**
    Go to the termination check
else
    Branching: branch on one of the fractional variables
    Go to the termination check
end if

number of locations by side and the total, including both sides. Finally, we indicate the ratio of the layout between sections and levels. These configurations will allow us to assess the impact of different aisle configurations. For each type of configuration, 10 sets of instances having each 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 requests were generated. For each of the five aisle configurations and for each of the 10 number of requests, we generated five instances. Thus, 50 instances were generated for each aisle configuration for a total of 250 instances. The typical operation of our industrial partner consists of orders containing fewer than 30 requests. We have generated instances with more requests to assess the performance of our methods.

The products are chosen randomly from the database obtained from our partner along with their specific volume and weight and placed randomly on the aisle. Our instances can contain more products to pickup than the total number of locations, meaning that products can be picked more than once.

Table 1: Types of aisle configuration

Type	Sections	Levels	Locations by side	Total locations	Ratio
1	12	4	48	96	3
2	24	4	96	192	6
3	36	4	144	288	9
4	48	4	192	384	12
5	60	4	240	480	15

We provide in Table 2 the average results for the method used by the company and for our algorithms on all 250 instances. The results are organized by the number of requests, thus each row presents the average of 25 instances for all five aisle configurations. The first column shows the number of pickups, followed by the solution value yielded by the company heuristic, then the solution of our ALNS algorithm, its running time in seconds, the best known solution yielded by CPLEX, its lower bound, percentage gap and running time in seconds. To validate our graph reduction method, we provide the results of the model for the cyclic and the reduced acyclic version.

It is clear that the ALNS algorithm is very effective and run in a few seconds even for the largest instances. Moreover, CPLEX, warm started by ALNS, is able to marginally improve the solution obtained by this algorithm. We observe an improvement for instances starting from 30 requests for both graph. Finally, as is known from other VRP problems, a branch-and-cut algorithm is capable of proving optimality or yielding tight optimality gaps for instances containing less than 100 nodes, as is corroborated by our experiments.

As it can be observed in the *Company* column of Table 2, which represents the current picking method, this heuristic never reaches optimality or yields solutions as good as those of the ALNS algorithm. In fact, these solutions are on average 5.76% worse than the solutions provided by our algorithms. In a large warehouse being operated daily over a long period, this represents a large reduction in the workload. Using the ALNS solutions to warm start CPLEX seems to be a good approach to help the solver in reaching optimality. The *Time* column shows how the ALNS is a fast heuristic given the large number of iterations we have allowed. With the branch-and-cut algorithm, we see that it was able to solve to optimality instances with up to 100 requests. We see that on average, the acyclic version is faster and obtains higher lower bounds. The use of an acyclic graph has reduced the overall average gap from 0.47% to 0.32% while being much more faster in terms of computation time.

In Table 3 we show the gap between the heuristic methods and the lower bound (LB) yielded by CPLEX after two hours of running time. We see that all instances with 50 requests or less were solved to optimality, and that the number of optimal solutions decreases when the number of requests goes up. In total, our cyclic model solved 164 instances and the acyclic model solved 189 instances to optimality from our set of 250 instances.

For the company heuristic, the gap with the best lower bound ranges between 0.28% and 6.98%. This shows that ALNS is able to yield better solutions, even for small instances. With respect to the best lower bound obtained by CPLEX, the ALNS provides an average gap of 0.34%. We note that these solutions were hardly improved by the exact methods.

Table 2: Average results ordered by the number of pickups

# of pickups		ALNS		Cyclic formulation				Acyclic formulation			
	Company	Sol.	Time (s)	UB	LB	Gap (%)	Time (s)	UB	LB	Gap (%)	Time (s)
10	282.88	282.08	0.24	282.08	282.08	0.00	0.04	282.08	282.08	0.00	0.00
20	517.84	509.84	0.28	509.84	509.84	0.00	0.00	509.84	509.84	0.00	0.04
30	692.24	674.48	0.60	673.92	673.92	0.00	22.80	673.92	673.92	0.00	0.80
40	889.76	861.60	1.20	860.40	860.40	0.00	101.52	860.40	860.40	0.00	11.48
50	1010.08	969.76	1.76	969.76	969.19	0.06	985.52	969.76	969.76	0.00	239.92
60	1147.52	1089.68	2.76	1088.56	1085.27	0.40	2463.28	1088.56	1087.05	0.20	1242.48
70	1395.76	1322.40	3.84	1321.68	1315.65	0.58	5086.28	1321.68	1317.01	0.35	3335.32
80	1529.12	1451.36	5.00	1451.04	1444.31	0.59	4401.96	1450.48	1446.04	0.35	3057.96
90	1738.64	1639.92	6.80	1639.76	1617.02	1.61	6344.48	1639.68	1625.96	0.85	5380.64
100	1886.96	1779.28	8.56	1778.64	1754.95	1.41	6153.68	1779.20	1755.19	1.43	5488.12
Average	**1109.08**	**1058.04**	**3.10**	**1057.57**	**1051.26**	**0.47**	**2555.96**	**1057.56**	**1052.73**	**0.32**	**1875.68**

Table 3: Average percentage gap of the heuristics and exact method

# pickups	Company	Gap (%) with respect to best LB	ALNS	Gap (%) with respect to best LB	B&C best LB	# optimals cyclic	# optimals acyclic
10	282.88	0.28	282.08	0.00	282.08	25	25
20	517.84	1.54	509.84	0.00	509.84	25	25
30	692.24	2.65	674.48	0.08	673.92	25	25
40	889.76	3.30	861.6	0.14	860.4	25	25
50	1010.08	3.99	969.76	0.00	969.76	22	25
60	1147.52	5.27	1089.68	0.24	1087.05	17	21
70	1395.76	5.64	1322.4	0.41	1317.01	8	15
80	1529.12	5.43	1451.36	0.37	1446.04	10	15
90	1738.64	6.48	1639.92	0.85	1625.96	3	7
100	1886.96	6.98	1779.28	1.35	1755.19	4	6
Average	**1109.08**	**4.16**	**1058.04**	**0.34**	**1052.73**	**164/250**	**189/250**

Table 4 shows that it is important to analyze the performance of the model against the various aisle configurations. Since the CNA-OPP is subject to movement restrictions, we must analyze the impact of the layouts configuration and the performance of the algorithms. With these constraints on section movements, we observe that different configurations can have a significant impact on the computation performance. The bigger it is, the more it can contain different products and have a longer distance for picking. In the same way, an instance with many requests in a small aisle, will contain the same products appearing multiple times. Therefore, it is difficult to compare the layout based only on the total distance. However, it is possible to measure the performance of methods for each type of aisle configuration based on the ratio between the number of sections and levels.

In Table 4 we present the average gap over five instances yielded by the solver after two hours, for each type of layout and all quantities of pickups, according to the aisle configuration created and presented in Table 1. When all instances are solved to optimality, we indicate it by a zero in bold. We can observe that the average gap (over 50 instance for each column) is well correlated to the ratio sections/levels outlined in Table 1. This means that for the same number of requests, optimal solutions are easier to obtain with aisles having more sections.

Even if the number of instances solved to optimality is the highest in the acyclic version of the model, the impact of the number of sections seems to be higher in the model with cycles. Both versions are still influenced by the number of sections in the aisle. Based on our algorithms, the easier aisle configuration is that with 60 sections. With the acyclic model on layout 5, we have obtained an average gap of 1.00% for all instances. With the cyclic model and the layout 5, this average gap is 1.35%.

For the acyclic version of the model, we notice that the layout 2 allows the model to perform well. It seems that the instances of this group with 90 requests were easier to solve than the instances from the layout 3. The layout 5 combined with an acyclic graph solves all instances with up to 70 requests to optimality. All other configurations are not

Table 4: Average percentage gap for each type of layout

# pickups	Cyclic model					Acyclic model				
	Layout 1	Layout 2	Layout 3	Layout 4	Layout 5	Layout 1	Layout 2	Layout 3	Layout 4	Layout 5
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
50	0.86	0.00	0.00	0.00	0.57	0.00	0.00	0.00	0.00	0.00
60	5.61	2.09	1.96	0.00	0.24	3.77	0.18	1.00	0.00	0.00
70	5.46	2.91	3.01	2.67	0.34	2.08	0.44	1.83	4.40	0.00
80	5.55	4.67	1.87	1.99	0.76	2.18	2.56	2.02	1.72	0.25
90	15.06	6.67	10.80	2.65	5.09	7.04	1.27	5.98	3.23	3.71
100	10.81	7.58	3.82	6.45	6.54	9.55	7.12	7.96	5.21	6.03
Average	**4.33**	**2.39**	**2.15**	**1.38**	**1.35**	**2.46**	**1.16**	**1.88**	**1.46**	**1.00**

able to solve to optimality all instances from the sample with more than 70 requests. Based on our set of instances, its is easier for CPLEX to solve problems in an aisle with more sections. The worst configurations for this group of instances is the first one, with only 12 sections. According to the movement restrictions, it seems difficult to create high quality tours in a short aisle. Obviously, a longer aisle will yield solutions with longer routes, but for which our methods can obtain solutions of better quality.

6 Conclusion

In this paper we have proposed a formulation for the CNA-OPP that yields a well-known CVRP. We have then been able to use classical CVRP algorithms to solve this difficult problem arising in warehousing operations. We have modeled the problem as a CVRP and based on the problem structure we have showed that the number of arcs can be drastically reduced leading to an acyclic model which is easier to solve. With a collaboration from an industrial partner, we have generated a large dataset of benchmark instances for the order picking problem based on a real data. We have proposed five configurations of an aisle and tested one heuristic and one exact algorithm against the picking method used by the company. We have shown that improvements of up to 4.16% can be obtained by using our methods.

References

[1] R. de Koster, T. Le-Duc, and K. J. Roodbergen. Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, 182 (2):481–501, 2007.

[2] M. Goetschalckx and D. H. Ratliff. Order picking in an aisle. *IIE Transactions*, 20 (1):53–62, 1988.

[3] J. Gu, M. Goetschalckx, and L. F. McGinnis. Research on warehouse operation: A comprehensive review. *European Journal of Operational Research*, 177(1):1–21, 2007.

[4] K. R. Gue, R. D. Meller, and J. D. Skufca. The effects of pick density on order picking areas with narrow aisles. *IIE Transactions*, 38(10):859–868, 2006.

[5] S. Hong, A. L. Johnson, and B. A. Peters. Batch picking in narrow-aisle order picking systems with consideration for picker blocking. *European Journal of Operational Research*, 221(3):557–570, 2012.

[6] J. Lysgaard. *CVRPSEP: A package of separation routines for the capacitated vehicle routing problem.* Institut for Driftøkonomi og Logistik, Handelshøjskolen i Århus, 2003.

[7] C. H. Mowrey and P. J. Parikh. Mixed-width aisle configurations for order picking in distribution centers. *European Journal of Operational Research*, 232(1):87–97, 2014.

[8] C.G. Petersen. An evaluation of order picking routing policies. *International Journal of Operations and Production Management*, 17(11):1098–1111, 1997.

[9] C.G. Petersen. The impact of routing and storage policies on warehouse efficiency. *International Journal of Operations & Production Management*, 19(10):1053–1064, 1999.

[10] J.-Y. Potvin and J.-M. Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3):331–340, 1993.

[11] H. D. Ratliff and A. S. Rosenthal. Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem. *Operations Research*, 31(3):507–521, 1983.

[12] K. J. Roodbergen and R. de Koster. Routing order pickers in a warehouse with a middle aisle. *European Journal of Operational Research*, 133(1):32–43, 2001.

[13] K. J. Roodbergen and I. F. A. Vis. A model for warehouse layout. *IIE Transactions*, 38(10):799–811, 2006.

[14] K. J. Roodbergen, G. P. Sharp, and I. F. Vis. Designing the layout structure of manual order picking areas in warehouses. *IIE Transactions*, 40(11):1032–1045, 2008.

[15] S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, 40(4):455–472, 2006.

[16] B. Rouwenhorst, B. Reuter, V. Stockrahm, G.J. Van Houtum, R.J. Mantel, and W.H.M. Zijm. Warehouse design and control: Framework and literature review. *European Journal of Operational Research*, 122(3):515–533, 2000.

[17] F. Semet, P. Toth, and D. Vigo. Classical exact algorithms for capacited vehicle routing problem. In P. Toth and D. Vigo, editors, *Vehicle Routing: Problems, Methods, and Applications*, volume 18. SIAM, Philadelphia, 2014.

[18] P. Shaw. A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*, 1997.